Claims

Please amend claims 1, 12, 13, 17, 24, 27, and 28, cancel claims 9, 10, and 14, and add new claims 33 and 34, as follows:

1. (**Currently Amended**) In a computer, a method of processing one or more software dependencies, the method comprising:

for one or more of the software dependencies, determining whether software associated with the software dependency is present on the computer;

responsive to determining the software associated with the software dependency is not present on the computer, acquiring, by a software package manager running on the computer, the software associated with the software dependency;

after acquiring the software associated with the software dependency, updating a database at the computer indicating the software associated with the software dependency is installed on the computer;

wherein at least one of the software dependencies specifies a plurality of software items forming a software package, wherein the software package comprises a mixture of native code components and platform-independent code components;

wherein at least one of the software dependencies refers to a list comprising one or more other software dependencies; and

wherein the acquiring the software associated with the software dependency comprises recursively processing the one or more other software dependencies.

2. (Original) The method of claim 1 wherein acquiring the software associated with the software dependency comprises acquiring a file comprising the list comprising one or more other software dependencies.

3. (Original) The method of claim 1 wherein acquiring the software associated with the software dependency comprises acquiring a list of one or more files from a remote location and acquiring the files in the list.

4. (Original) The method of claim 1 wherein one or more of the software dependencies is associated with a location whereat the list of other software dependencies can be found.

5. (Previously Presented) The method of claim 1 wherein the database is operable to indicate whether a plurality of software components are installed via a single name associated with the plurality of software components.

6. (Previously Presented) The method of claim 1 wherein the database is operable to indicate whether a plurality of software dependencies are installed via a single name associated with the plurality of software dependencies.

7. (Original) The method of claim 1 wherein:

one or more dependencies in the list of software dependencies is associated with a version number; and

determining the dependency is not present on the computer comprises determining software satisfying the version number is not present on the computer.

8. (Original) The method of claim 7 wherein at least two software dependencies are associated with different version numbers.

9. (**Canceled**)

10. (**Canceled**)

11. (Original) The method of claim 1 wherein acquiring dependencies is deferred until execution of software associated with the dependencies is requested.

12. (Previously Presented) A computer-readable medium comprising computer-executable instructions for performing at least the following to process one or more software dependencies in a computer:

for one or more of the software dependencies, determining whether software associated with the software dependency is present on the computer;

responsive to determining the software associated with the software dependency is not present on the computer, acquiring, by a software package manager running on the computer, the software associated with the software dependency; and

after acquiring the software associated with the software dependency, updating a database at the computer indicating the software associated with the software dependency is installed on the computer;

wherein at least one of the software dependencies specifies a plurality of software items forming a software package, wherein the software package comprises a mixture of native code components and platform-independent code components;

wherein at least one of the software dependencies refers to a list comprising one or more other software dependencies; and

wherein the acquiring the software associated with the software dependency comprises recursively processing the one or more other software dependencies.

13. (**Currently Amended**) In a computer, a method of specifying a software dependency, the method comprising:

specifying a name of the software dependency, wherein the name is operable to identify a list of one or more other software dependencies;

specifying a version of the software dependency;

comparing the version for the software dependency against a version of software installed at the computer; and

responsive to determining the version installed at the computer is not sufficient, acquiring and installing, by a software package manager running on the computer, the software dependency, wherein the acquiring and installing the software dependency comprises recursively specifying the one or more other software dependencies**;**

wherein the name of the software dependency is associated with a software package comprising a plurality of software components, wherein the plurality of software components comprises a mixture of native code components and platform-independent code components.

14. (**Canceled**)

15. (Original) The method of claim 13 wherein the software dependency is associated with a software package depending on at least one other software package.

16. (Canceled)

17. (**Currently Amended**) In a computer, a method of processing a name designating software, the method comprising:

consulting a database to see if software associated with the name is already installed at the computer;

responsive to determining software associated with the name is not already installed at the computer, acquiring, by a software package manager running on the computer, the specified software; and

responsive to determining software dependencies associated with the specified software are not already installed at the computer, acquiring, by the software package manager running on the computer, the software dependencies;

wherein the name is operable to specify a plurality of software components, **wherein the name is associated with a software package comprising the plurality of software components, wherein the plurality of software components comprises a mixture of native code components and platform-independent code components,** and wherein acquiring the specified software comprises recursively processing software dependencies associated with the name to find one or more other software dependencies associated with names designating software.

18. (Original) The method of claim 17 wherein the plurality of software components is divided among a plurality of files.

19. (Original) The method of claim 17 wherein the plurality of software components comprises a plurality of files.

20. (Original) The method of claim 17 wherein

the name is operable to specify a plurality of different software components; and

different versions are associated with the different software components.

21. (Canceled)

22. (Original) The method of claim 17 wherein the name is operable to specify a list of one or more software dependencies, the method further comprising:

for at least one software dependencies, determining software associated with the dependency is already installed at the computer, wherein the dependency specifies a plurality of software components.

23. (Original) The method of claim 17 wherein acquiring the specified software comprises identifying an other name associated with a plurality of software components, and acquiring the plurality of software components associated with the other name.

24. (**Currently Amended**) A computer-readable medium comprising a computer software package of a nestable software package format, wherein the software package format comprises:

a package name; and

a list of dependencies, wherein the list of dependencies is operable to specify one or more other software packages on which the software package depends, wherein at least one of the other software packages is associated with another package name and another list of dependencies and is also of the nestable software package format**, and wherein at least one of the other software packages comprises a plurality of software components, wherein the plurality of software components comprises a mixture of native code components and platform-independent code components**;

wherein the nestable software package format is operable to specify a remote location from which one or more components required by the software package can be acquired by a software package manager running on a computer, and the nestable software package format is operable to be recursively processed by recursively processing the list of dependencies before finishing installation of the software package.

25. (Original) The computer-readable medium of claim 24 wherein the nestable software package format is operable to specify one or more components required by the software package.

26. (Previously Presented) The computer-readable medium of claim 24 wherein the nestable software package format is operable to specify a remote location from which one or more components required by the software package can be acquired.

27. (**Currently Amended**) A computer-readable medium comprising a software distribution package for installing software at a computer, wherein the software distribution package comprises:

~~one or more~~ **a plurality of** items for installation at the computer**, wherein the plurality of items comprise a mixture of native code components and platform-independent code components**; and

a dependency list indicating one or more items depended on by the software, wherein at least one of the items on the dependency list is not contained in the package, and the software package indicates a remote location from which the item can be acquired and installed by a software package manager running on the computer;

wherein the items are specified in the dependency list by a name operable to specify a plurality of additional items, and wherein the dependency list is recursively processed.

28. (**Currently Amended**) A computer system for executing a software package comprising a specified list of one or more dependencies, the computer system comprising:

a database indicating the installation status of one or more software components; and

a software package manager running on the computer system operable to resolve the specified list of one or more dependencies by consulting the database to determine whether a dependency is installed and further operable to acquire a dependency determined as not installed**, wherein at least one of the dependencies is associated with a software package comprising a plurality of software components, wherein the plurality of software components comprises a mixture of native code components and platform-independent code components**;

wherein the software package manager is operable to process at least one item in the specified list of dependencies referring to an other list of dependencies and process the specified list of one or more dependencies recursively.

29. (Original) The computer system of claim 28 wherein the software package manager is further operable to acquire a file at a remote location indicating the other list of dependencies.

30. (Original) The computer system of claim 28 wherein the other list of dependencies is specified via an URL.

31. (Original) The computer system of claim 28 further comprising a browser;

wherein the software package manager is operable to initiate execution of a software package as directed by the browser upon encountering HTML tags indicating the specified list of dependencies.

32. (Previously Presented) The method of claim 1 further comprising:

after acquiring the software associated with the software dependency, installing the software.

33. (**New**) The method of claim 1, further comprising:

after processing the one or more software dependencies, uninstalling the software associated with the one or more software dependencies based on the database.

34. (**New**) The method of claim 1 wherein the mixture of native code components and platform-independent code components are grouped together in a distribution unit.